TECHNICAL UNIVERSITY OF CRETE

THESIS

Employing Hypergraphs for Efficient Coalition Formation with Application to the V2G Problem

Author:
Filippos CHRISTIANOS

Supervisor:
Georgios CHALKIADAKIS

A thesis submitted in fulfillment of the requirements for the degree of Electronic Engineer and Computer Science

in the

Department of Electrical and Computer Engineering
Technical University of Crete

November 6, 2016

"Lorem ipsum dolor sit amet, consectetur adipiscing elit."

Lorem Ipsum

TECHNICAL UNIVERSITY OF CRETE

Abstract

Technical University of Crete
Department of Electrical and Computer Engineering

Electronic Engineer and Computer Science

Employing Hypergraphs for Efficient Coalition Formation with Application to the V2G Problem

by Filippos CHRISTIANOS

This paper proposes, for the first time in the literature, the use of *hypergraphs* for the efficient formation of effective coalitions. We put forward several formation methods that build on existing hypergraph pruning, transversal, and clustering algorithms, and exploit the hypergraph structure to identify agents with desirable characteristics. Our approach allows the near-instantaneous formation of high quality coalitions, adhering to multiple stated quality requirements. Moreover, our methods are shown to scale to *dozens of thousands* of agents within fractions of a second; with one of them scaling to even *millions* of agents within seconds. We apply our approach to the problem of forming coalitions to provide *(electric) vehicle-to-grid (V2G)* services. Ours is the first approach able to deal with *large-scale*, *real-time* coalition formation for the V2G problem, while taking *multiple criteria* into account for creating the electric vehicle coalitions.

Acknowledgements

The acknowledgments and the people to thank go here, don’t forget to include your project advisor…

Contents

List of Figures

List of Tables

List of Abbreviations

CF	**C**oalition **F**ormation
CS	**C**oalition **S**tructure
CSF	**C**oalition **S**tructure **G**eneration
EV	**E**lectric **V**ehicle

List of Symbols

a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

Chapter 1

Introduction

One domain where the formation of coalitions comes naturally into play is the so-called *vehicle-to-grid (V2G)* problem. In V2G, battery-equipped *electric vehicles (EVs)* communicate and strike deals with the electricity Grid in order to either lower their power demands, or return power back to the network when there is a peak in the request for power. This helps the Grid to maintain a balanced power load [13]. G2V is V2G's "sister" problem, where EVs connect and draw power from the Grid without overloading it [16]. In both cases, the coordination of EVs efforts, is essential.

As such, several recent approaches have called for the formation of EV coalitions in order to tackle the V2G problem [11, 10, 9]. The existing approaches, however, typically exhibit the following characteristics: *(a)* they attempt to form *optimal* coalitions or coalition structures; and *(b)* they either attempt to form coalitions with respect to a single criterion, or employ lengthy negotiation protocols in order to capture various coalitional requirements while respecting the constraints of individual agents.

The inherent hardness of the optimal coalition structure generation problem [12], however, and the fact that negotiation protocols can be lengthy and thus highly time consuming, severely restricts the practicality and scalability of such algorithms: they can handle at most a few hundred EVs. In reality though, there exist hundreds of thousands of EVs that connect to the Grid, and could potentially offer their services; any formed coalition would be required to possess *a multitude of desirable characteristics* (e.g., high collective storage capacity, and high collective discharge rate); and, if the aim is to balance electricity demand in real time, any such service should be offered by the appropriate coalition almost instantaneously.

In this paper, we overcome the aforementioned difficulties by employing, for the first time in the literature, *hypergraphs* to achieve the timely formation of coalitions that satisfy *multiple criteria*[1]. In our approach, EV agents that share specific characteristics are organised into *hyperedges*. Then, building on the existing hypergraphs literature [6, 19], we propose algorithms for *(i)* hypergraph *pruning*, to focus on interesting parts of the search space; *(ii)* hypergraph *transversal* to identify sets of vertices (agents) that combine several desirable characteristics; and *(iii)* hypegraph *clustering*, that allows the identification of clusters of high quality agents. Moreover, we put forward *(iv)* a heuristic formation algorithm that benefits from pruning and generates high quality coalitions near-instantaneously, while scaling linearly with the number of agents.

In contrast to existing approaches, we do not attempt to generate an optimal coalition structure, nor do we attempt to compute a single optimal

[1]A sketch of these ideas appeared in a short ECAI-2016 paper [5].

coalition. Instead, we exploit the hypergraph representation of our problem in order to select agents and form highly effective coalitions, while being able to scale to *dozens of thousands* of agents within fractions of a second; and, in the case of our heuristic method, even to *millions* of EV agents in seconds.

Though here we apply it to the V2G problem, our approach is generic and can be used in *any* coalition formation setting. It is perhaps surprising that a powerful model like hypergraphs has not been so far exploited for devising efficient coalition formation methods, despite its intuitive connections to the concept of coalitions. Regardless, we are not aware of any work to date that has exploited hypergraphs and related algorithms in order to perform *real-time*, *large-scale*, *multi-criteria* coalition formation, as we do in this paper.

1.1 Coalition Formation

Coalition formation (CF) is a paradigm widely studied in multiagent systems and economics, as means of forming teams of autonomous, rational agents working towards a common goal [2]. Individual agents usually have different degrees of efficiency. Thus, we must form groups of agents with characteristics that compliment each other and exploit their individual strengths[15]. As discussed in [14], coalition formation has three activities. *Coalition structure generation*, being the first, is the partitioning of the set of agents into mutually disjoint coalitions (or groups), in a way that the resulting coalitions maximize the sum of the rewards of all agents (known as social welfare) [12]. Next is *the optimizations problem* of each coalition, that tries to maximize the rewards from outside the coalition and optimize the allocation of resources and tasks between agents of the respective coalition. The last activity of CF, is the *division of the rewards* among agents. This must be done in such a way that the rewards are fair, and no agent can be motivated to leave his coalition.

Finding the optimal coalition structure is generally computationally expensive, especially in a large set of agents, and the computational requirements grow exponentially. As such, finding in a respectable time a CS that is within a bound of the optimal one, is also hard problem. There are several attempts to solve this problem [14] [12].

Nevertheless in this thesis, we will not attempt to solve the CSG problem. Instead we will focus on finding a simple coalition that is able to perform a specific task. Such a coalition will be created by selecting agents from an extensive set, in a way that the result can efficiently handle the appointed task. In addition each agent will have several attributes that contribute in different ways to the completion of the goal. We will not be using a utility function (that ultimately combines the attributes, losing accuracy). That will be called *multi-criteria* coalition formation. Due to the nature of this problem, the results cannot be easily evaluated. There is possibly a great number of possible coalitions with similar capacity to handle the task. In addition, since the agent set is magnitudes larger that what usual optimal CF algorithms can handle we cannot find how close to optimality our solution is. Thus we will focus on creating coalitions that can complete the task efficiently and can be generated in a minimal amount of time.

This problem can be quite natural in todays world. There are several real world examples where efficiency is sacrificed for performance. In this thesis, we will present a way to form an EV coalition in just a few seconds, able to fulfill the energy requirements of the Smart Grid.

Chapter 2

Chapter Title Here

2.1 Related Work

Here we review related work, and highlight its differences to our approach. To begin, Valogianni *et al.* [16] propose an *adaptive smart charging algorithm* that adjusts *the power drawn* from the Grid for charging EVs, based on each EV owner's utility from charging. The approach employs *reinforcement learning* for capturing agent needs and behaviour; and an optimization module schedules the charging of each EV to maximise its utility subject to network constraints. Though effective, it does not focus on the problem of feeding the network with power drawn from EVs in a coordinated fashion, and as such there is no mentioning of EV coalitions in that work.

By contrast, an attempt to explicitly sell power in the regulation market via the formation of EV coalitions is presented in [10]. In that work, EV coalitions provide the following service to the Grid every few seconds: they either *scale down* their power draw (or discharge); or they *scale it up*, and request more power. The approach is quite effective, but there is a need for a complicated EV selection process by an aggregator agent, and simulations presented in that paper involved a pool of 300 vehicles only.

A paper adopting a game-theoretic perspective on the formation of coalitions in the Smart Grid is [17]. It constitutes an attempt to solve the optimal coalition structure generation problem (CSG). Forming *virtual energy consumer* coalitions, manages to flatten the demand in order to get better prices in what could be a G2V arrangement. By solving the CSG, it finds the best VEC for every consumer on the market; and guarantees a core-stable payoff distribution outcome. Unfortunately, this solution is shown to work on social graphs with only a handful of agents. By contrast, our approach manages to produce high quality solutions in milliseconds, and scales to the millions.

Two recent papers which study *cooperative games* defined *over graphs* that impose constraints on the formation of the coalitions, are [3] and [4]. Specifically, they assume that the environment possesses some structure that forbids the creation of individual coalitions, due to limited resources and existing physical or even legal barriers. This is captured by an undirected graph providing a path connecting any two agents that can belong to the same coalition. Both of these papers, however, do not employ hypergraphs in any way. Hypergraphs have in fact been used for modelling agent interactions in cooperative game settings where agents can simultaneously belong to multiple coalitions [8] [20]. Nevertheless, all of these papers [3] [4] [8] [20] focus on studying the theoretical problem of achieving *coalitional stability* via appropriately distributing the payoff among the agents; rather than providing algorithms for large-scale coalition formation in real-world settings, as we do in this work. By contrast, two papers that study the generation of optimal coalition structures while focusing on stability are [1] [18]. These approaches scale to thousands of agents - but not to millions, as ours (which does not form optimal coalitions), and do not tackle multiple formation criteria.

A paper that is more related to our work here, in the sense that it exploits constraints among vehicles for coalition formation, is the work of Ramos *et al.* [11]. They propose the dynamic formation of coalitions among EVs so that they can function as *virtual power plants* that sell power to the Grid. However, that work also attempts to tackle the optimal CSG problem. The method relies heavily on a inter-agent negotiations protocol; and is empirically shown to produce solutions that are close to optimal (98%), but this is when tested in scenarios with a few dozens of agents only. Moreover, there is only a single criterion for the formation of a coalition—namely, the *physical distance* among the EVs. Physical distance, however, is not a very natural criterion; and, in any case, it is imperative that a multitude of criteria is taken into account—such as capacity, discharge power, and perceived reliability (see, e.g., [9]). Our approach, by contrast, is able to take into account any number of natural criteria to form EV coalitions.

Chapter 3

Chapter Title Here

3.1 Our Approach

In order to develop multi-criteria coalition formation algorithms that generate coalitions efficiently, we employ the concept of *a hypergraph*. A hypergraph $H = (V, E)$ is a generalization of a graph, where each *hyperedge* $e \in E$ can contain any number of *vertices (or nodes)* in the set V. Vertices in H correspond to agents; while we view a hyperedge as corresponding to some particular *attribute* or *characteristic* possessed by the agents in the hyperedge. In the V2G setting, the agents correspond to EVs (i.e., an EV is represented by a node in our hypergraph); while the hyperedges correspond to vehicle characteristics. More specifically, a hyperedge corresponds to a "quality level" of some EV attribute, as we explain below.

In order to represent the different *quality* of the various hyperedges, and utilize it in our algorithms, we mark each hyperedge with a weight.[1] These weights define the *degree* of a node: *The degree $deg(u)$ of a node u is the sum of the weights of its edges*. Intuitively, *a high degree node is a high quality one*. This fact is exploited in our algorithms below. A hyperedge (of a given quality) will be also called a *category*. The (quality of the) categories to which an EV belongs will be influencing the decisions of our *hypergraph pruning* algorithm, which we describe in Section 3.1.2 below. A node that belongs to a hyperedge characterizing the quality of a given agent attribute, cannot belong to some other hyperedge characterizing the quality of the same attribute.

[1]In our implementation, the weight of the edges, according to the quality of each attribute(capacity, reliability and discharge), are as follows: {*extremely-high: 8, very-high: 7, high: 6, medium-high: 5, medium-low: 4, low: 3,very-low: 2, extremely-low:1*}. Thus we have 24 edges + 1 containing commitment of EVs.

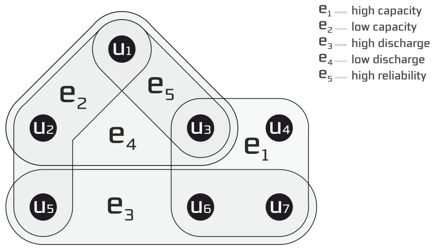

FIGURE 3.1: A simple hypergraph

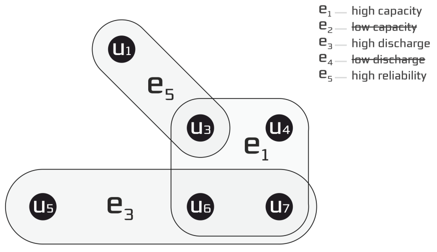

FIGURE 3.2: Pruning a simple hypergraph

To illustrate the use of hypergraphs in our setting, consider for example the hypergraph of Fig. 3.1, which contains the hyperedges $e_{1...6}$ and vertices $u_{1...7}$. It is clear in this example that vertices may belong to multiple hyperedges: the hyperedge e_1 contains the vertices $u_{3,4,6,7}$, while the vertex u_1 belongs in the hyperedges e_2, e_5, e_4. Vertices in Fig. 3.1 correspond to EVs; while the hyperedges correspond to the "quality" of the following EV attributes: *capacity*, *discharge rate* and *observed reliability*. The meaning of these attributes is intuitively straightforward, but will be nevertheless explained in Section 3.1.1 below. Each attribute is related to at least one hyperedge in the hypergraph. For instance, in Fig. 3.1, the *capacity* attribute is represented by three hyperedges in the hypergraph: *low-capacity*, *medium-capacity*, and *high-capacity*. As noted above, no node can belong in more than one capacity-related hyperedges. In our figure,

- the hyperedge $e1$ represents the nodes which have high capacity;
- the hyperedge $e2$ contains nodes that have low capacity;
- $e3$ and $e4$ include the vehicles with high and low discharge rate, respectively;
- finally, $e5$ contains nodes that are expected to the *highly reliable*.

For example, node $u1$ is a *low-capacity*, *low-discharge* but *highly reliable* vehicle—while node $u3$ is a *high-capacity*, *low-discharge* and *highly reliable* one.

Organizing the information relating to specific agent attributes using hyperedges, enables us to both access this information efficiently, and keep it organized. Moreover, in many settings, agent characteristics captured by hyperedges, naturally correspond to criteria according to which we can form coalitions. For example, it is conceivable that we want to use agents with *high capacity* from the respective *high-capacity* edge, if our goal is to form coalitions with *high capacity*. Our approach of using hypergraphs is even more generic than what implied so far, since we can easily define hyperedges that contain agents which are or are not *permitted* to connect with each other, for various reasons; and since we can exploit the hypergraph to allow the formation of coalitions according to a multitude of criteria.

3.1.1 Criteria for Forming Coalitions

The algorithms presented in this work can be employed by any entity or enterprice (such as the Grid, utility companies or Smart Grid cooperatives) that wants to form EV coalitions for the V2G problem, using any set of criteria of its choosing. Here we identify three such natural criteria, namely *reliability, capacity* and *discharge rate*. These formation criteria are consistently mentioned in the related literature, though perhaps not with these exact names, and not explicitly identified as such [10, 9, 16].

First of all, a coalition has to be consistently *reliable*, i.e. it should be able to serve the power that has been requested without any disruptions. For a coalition to be reliable, its members must be reliable too, and gaps in reliability must be met with backup agents. We define *agent reliability* as *the estimated probability that an agent will fulfill its promises*. The *promise* of an agent is its *commitment* on being connected to the Grid during a specific time slot in order to contribute via providing energy to the Grid, if so requested. Such slots naturally correspond to electricity trading intervals.

In addition, a coalition must fulfill a *capacity* requirement. The *capacity* of a coalition is the amount of electricity (measured in kWh) the coalition will be offering to the Grid; while the capacity of en EV is, similarly, the amount of electricity (in kWh) the EV will be offering to the Grid. In fact, gathering enough EV capacity to cover the Grid needs during high demand periods, is the main objective of any V2G solution.

Another factor in the V2G problem is the *discharge rate* of a coalition (or, of a single EV)—the rate by which the coalition (resp., the EV) is able to provide (electrical) energy to the Grid over a specified time period. Discharge rate is measured in kW. A high coalitional discharge rate could be required in cases where capacity should be offered within a small amount of time, for example when the Grid is under a heavy demand load. Naturally, a coalition has a high discharge rate if its members discharge rates are high; for our purposes, we assume that the discharge rate is additive, i.e., the discharge rate of a coalition is the sum of its EVs discharge rates. In Section 4.1, we will be forming coalitions in order to meet specific capacity and discharge rate targets; and observing how reliable the coalitions meeting these targets are.

Now, the hypergraph used in our current implementation was designed so that it could easily satisfy requests pertaining to these particular criteria. As such, there was a total of 25 hyperedges in the hypegraph—{*extremely-high, very-high, high, medium-high, medium-low, low, very-low, extremely-low*} $\times$ {*capacity, discharge rate, reliability*}; and a *committed* one, containing EVs that have stated they will be connecting to the Grid during the particular slot.

In our model, we assume that, at any time step that this is required—due to a consumption peak, an unplanned event, or the need to regulate frequency and voltage—the Grid (or some other entity) advertises its demand for a V2G coalition with several desirable characteristics. As noted in [9], individual EVs are well-suited for providing services at short notice. What we show in this paper, is that we can select agents from a huge pool of EVs to form *coalitions* that are able to provide large amounts of power at short notice, and with high reliability.

3.1.2 Pruning the Hypergraph

An important aspect of using hypergraphs for dealing with large state-spaces, is the resulting ability to perform node and edge pruning. Since dozens or hundreds of thousands of our EVs populate the hypergraph, and each one is a member of several hyperedges, running the algorithms without pruning would require an enormous amount of computing power. However, due to the nature of the hypergraph, and the way we store our vehicles and their attributes, it is extremely easy and effective to narrow down the number of vehicles and edges used, by leaving out EVs that are less promising as coalition members. For example, if achieving a high capacity for the to-be-formed coalition is a key goal, then, intuitively, we can narrow down our search for coalition members by focusing only on nodes belonging to the set of hyperedges (or "categories") $highcapacity \cup veryhighcapacity \cup exhighcapacity$.

To illustrate pruning, Fig. 3.1 shows a hypergraph that contains all EVs. In order to reduce the size of the hypergraph and thus the computing requirements, we could keep only EVs belonging to at least one high quality edge, as shown in Fig. 3.2.

Algorithm 1 Pruning the Hypergraph

```
1: procedure PRUNING($H, CategoriesKept$)
2:     for Hyperedge ∈ H do
3:         if $Hyperedge \in CategoriesKept \cap Committed$ then
4:             $NewHEdges \leftarrow NewHEdges \cup HyperEdge$
5:             $NewNodes \leftarrow NewNodes \cup HyperEdge.nodes$
6:         end if
7:     end for
8:     $NewHGraph \leftarrow Hypergraph(NewNodes, NewHEdges)$
9: end procedure
```

Algorithm 1 is our implementation of pruning. The algorithm iterates over all hyperedges in the given hypergraph H, and keeps only the nodes belonging to hyperedges that correspond to the specified "categories of interest" (*CategoriesKept* in Alg. 1).

In our implementation, the *CategoriesKept* are heuristically selected, and depend on the algorithms. For instance, the *minimal transversal* algorithm requires a more aggressive pruning, since its complexity is sensitive to the number of nodes used as input (cf. Section 3.1.3), and we therefore empirically feed it with as few hyperedges as possible.

Our experimentation indicates that the use of pruning can lead to a significantly smaller hypergraph, and to vast improvements in terms of execution time for our algorithms. In our simulations, the hypergraphs are pruned to about $1/20$ of the initial size of the EVs pool, without sacrificing the methods' performance (cf. Section 4.1.1). Moreover, pruning using Algorithm 1 is almost instantaneous.

3.1.3 Minimal Transversal Algorithm

Using hypergraphs allows to use an intuitive approach for locating agents for coalitions: to generate the set of *minimal transversals* for the *high-value hyperedges* [6]. A *transversal* (or *hitting set*) of a hypergraph H, is a set $T \subseteq V$ with hyperedges X where $X = E$ (i.e., vertices in T belong to *all* hyperedges in E). A *minimal transversal* is a set that does not contain a subset that is a hitting set of H. As such[2], generating several minimal transversal sets for *high-quality* hyperedges is expected to identify agents which are high-quality and should be used in the formation of a coalition. Subsequently, we join those agents together until our criteria are met.

Our approach with the minimal transversal set is to prune all edges but those of extremely high quality that are also "committed", as seen in Algorithm 2. Then we generate progressively the minimal hitting sets, using an algorithm similar to [6]. That is, we first generate the minimal hitting sets containing one node, then those with two, and so on. Then we randomly

[2]Of course there can be more than one minimal transversals, and it is not necessary that they have the same cardinality.

pick agents belonging to those minimal transversals, until the coalitions requirements are met. If the requirements are met during the progressive minimal transversal generation process, no further minimal transversals are generated.

To illustrate this concept with the help of Fig. 3.1, we prune the hypergraph to keep only the high-quality edges e_1, e_3, e_5, leaving us with the nodes $u_1, u_3 ... u_7$ and edges e_1, e_3, e_4, as seen in Fig. 3.2. Then we generate all the minimal transversal sets. The minimal transversals generated first are the ones with two nodes (since there are no minimal transversals with one node) i.e. the following$\{u_3, u_5\}, \{u_1, u_7\}, \{u_6, u_1\}$.

This method creates a set of agents with uniformly distributed high-quality characteristics. Though this is desirable in theory, in practice the results vary depending on the generated minimal transversal set. There are characteristics which might be of higher importance than others and this cannot be taken into account by the transversal algorithm due to its nature. Regardless, this method could be of much use for creating a base of quality agents; for uniformly improving the quality of an already formed coalition by adding agents from the minimal transversal sets; and for creating versatile coalitions without focusing on specific attributes.

Algorithm 2 Coalition formation using Minimal Transversal

```
1: procedure MINIMALTRANSVERSAL($H$)
2:     $H \leftarrow Prune(H, exhigh)$    ▷ exhigh signifies all hyperedges with exhigh qualities
3:     $T = \emptyset, C = \emptyset$    ▷ Start with an empty coalition
4:     for i=1 to $|E|$ do ▷ where $|E|$ is the number of edges in the (pruned) $H$
5:         Create the union $U$ of minimal transversal sets with size $i$, generated from $H$.
6:         $T = T \cup U$
7:         while $C$ does not meet the criteria do
8:             Randomly select an unselected node $\in T$ and add it to $C$
9:         end while
10:        if criteria have been met then
11:            return formed coalition $C$
12:        end if
13:    end for
14: end procedure
```

Line 6 of Algorithm 2 is our implementation of minimal transversal [6]. Though there is no known polynomial time algorithm for the general hypergraph transversal problem, the algorithm given was shown experimentally to behave well in practice, and its memory requirements are polynomially bounded by the size of the input hypergraph, though it comes without bounds to its running time.

3.1.4 Clustering Algorithm

The second approach is to create clusters of agents. After creating said clusters, we efficiently calculate the best cluster and then sample EVs from that group until our coalition criteria are met.

In more detail, we first generate a hypergraph of EV agents with the characteristics described previously. Then, hypergraph clustering is performed. The hypergraph clustering itself is an implementation of that proposed in [19], and is conducted as follows.

We begin by implementing functions that calculate

- *the Incidence Matrix*: A matrix H with entries $h(u, e) = 1$ if $u \in e$ and 0 otherwise.
- *the Weight Matrix*: A diagonal matrix W containing the weights of the hyperedges.
- *D_u and D_e*: Matrices containing the node and hyperedge degrees respectively.
- *the Adjacency Matrix*: A matrix defined as $A = HWH^T - D_u$

The matrices above are used for the final calculations of the hypergraph *Laplacian matrix*. This a matrix representation of a graph, that has information on the degrees of the nodes, and their connections with the hyperedges (cf. [19], Section 5).

As explained in [19], having the Laplacian, enables us to calculate the Φ eigenvectors $[\Phi_1 ... \Phi_k]$ corresponding to the k lowest eigenvalues. These can then define $X = [\Phi_1 ... \Phi_k]$, a matrix that can be employed for k-way partitioning to cluster our agents. This is achieved via running the *k-means* algorithm [7] on the row vectors of X[19]. As explained in [19], the rows of X are representations of the hypergraph vertices in the k-dimensional Euclidean space. Of course, choosing a value for k has to be decided empirically. In Section 4.1.4 we will be testing different values for k. After generating the clusters, we are given the task to locate the "best" cluster among them. To do this efficiently, we simply sort them by looking at *the average of the node degrees*. This provides us with a cluster that is better than the rest. We then sample nodes from the best cluster until our criteria are met. Algorithm 3 summarizes the method.

Algorithm 3 Coalition formation using Hypergraph Clustering

```
1: procedure CLUSTERING($H$)
2:     $H \leftarrow Prune(H, (vhigh \cup exhigh))$    ▷ exhigh and vhigh signify the
   sets of extremely high and very high quality hyperedges respectively
3:     Generate k clusters using the algorithm described in 3.1.4 [19]
4:     $C = \emptyset$                                  ▷ Start with an empty coalition
5:     Find the best cluster, $A$, by comparing the sum of node degrees of
   each cluster.
6:     while $C$ does not meet the criteria do
7:         Randomly select a node $\in A$ and add it in $C$
8:     end while
9: end procedure
```

3.1.5 A Heuristic Algorithm

While using a minimal transversal generates quality sets of agents, computing the *degree* of a node can identify single agents with many quality attributes. As an example, when we have a reliable coalition as a base but we require more capacity, we can use the sorted list we have generated, to pick agents with high capacity. Intuitively, this approach will result to picking high overall quality agents for our coalition. We can also create coalitions by using only the best available agents. Moreover, we can use the aforementioned sorted-by-degree list of nodes in order to "fill gaps" and improve on the quality of already formed coalitions.

Thus, our heuristic method operates as follows. *(1)* First, we prune the hypergraph to include only “promising” nodes and hyperedges. For instance, we exclude nodes not in *extremely high* or in *very high* hyperedges. *(2)* Then we sort the remaining nodes based on their node degree. *(3)* Finally, we pick the highest degree nodes from the list until the coalition criteria are met. By starting at the top of the list, we can guarantee that agents have many positive characteristics.

We can see at step *(1)* above, that this algorithm, like the rest of our methods, employs pruning. As such, it does exploit the hypergraph structure. However, in practice the algorithm can deliver excellent results without much pruning. In our experiments in Section 4.1 below, the heuristic approach is shown to outperform the rest while pruning only the noncommitted nodes in the hypergraph. In fact, one strength of this approach is that it does not *rely* on pruning, since its complexity is low: essentially, that of the algorithm employed for sorting (i.e., $O(nlogn)$, since we use with Python’s built-in *Timsort* algorithm). By not relying on pruning, the algorithm can focus on promising nodes with high node degree (and, therefore, quality), irrespective of the exact hyperedges to which they belong.

3.1.6 A Simple Sampling Method

For interest, and in order to have a benchmark for the rest of our algorithms, a simple sampling algorithm was also developed. The algorithm takes random samples until the specified goals are achieved.

Chapter 4

Chapter Title Here

4.1 Experiments and Results

In this section we present the evaluation of our algorithms. First we explain how the EV population is generated, and the time this generation process takes. Then, the performance of the algorithm is evaluated in terms of the quality of the formed coalition and also in terms of execution time and scaling behavior. All figures and tables present average values over multiple runs. Specifically, we generated 20 hypergraphs with $20,000$ EVs each, and then ran each algorithm on every hypergraph 10 times, and took the averages (and the average of those averages). Our experiments were run on a Sandy Bridge i7-2600K at 4.2 GHz. All the tests were running on a single thread on Python, meaning that there is a lot of room for optimization.

4.1.1 Generating the EV Population

To generate the population for each type of experiment we create the vehicles one by one, by first generating its properties as follows. The capacity of each vehicle is generated from a Gaussian distribution with mean value 100 and $\sigma = 80$. The discharge rate of each vehicle is generated from a Gaussian distribution with mean value 10 and $\sigma = 5$. The reliability of each vehicle is picked from a Gaussian distribution with mean value 0 and $\sigma = 1$. Each EV's commitment of being connected to the Grid is a *true* / *false* variable, with a 0.9 probability of being *true*. If *true*, then the EV is inserted in the *committed* hyperedge. When a vehicle has its properties created, it is added in the pool of available EVs. The computational complexity of generating the hypergraph is, as expected, $O(n)$.

The coalition requirements are set to values which are commonly used in the regulation market [9], namely the following two. First, each coalition must have a total capacity of at least $10MWh$. The discharge rate must also be at least $1MW$ [9] These values are kept constant throughout all experiments—except when we test scaling against an increasing capacity goal, where capacity is treated as a variable.

Creating the hypergraph is a problem that scales linearly with time. Specifically, generating the hypergraph, including the vehicles and distributing them to hyperedges, takes a very small amount of time and scales linearly up to a million within a minute (Table 4.2). As mentioned above, the initial EV population was $20,000$ nodes. However, before feeding the nodes to the algorithms, we pruned the hypergraph to keep promising nodes. Table 4.1 shows the average hypergraph size finally fed to the algorithms.

Algorithm	Nodes after Pruning	Edges after Pruning
Transversal	1148.4	4
Clustering	1218.8	7
Heuristic	18012.6	25

TABLE 4.1: Pruning Results

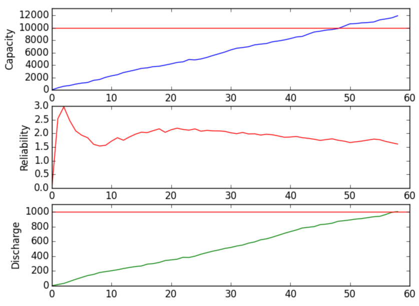

FIGURE 4.1: Coalition formation with the Heuristic Algorithm

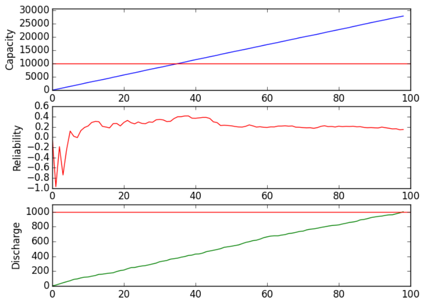

FIGURE 4.2: Coalition formation with the Clustering Algorithm

4.1.2 Forming the Coalitions

We now proceed to evaluate the performance of our algorithms. Our evaluation will examine *(a)* how fast and *(b)* by selecting how many vehicles they can meet the set requirements. Naturally, the faster an algorithm forms a coalition that meets all the requirements, the better. Moreover, coalitions with fewer vehicles are preferable, since intuitively, this allows for a more efficient allocation of resources, and also means that fewer EVs will share the payoff associated with forming the coalition (exactly how this payoff allocation will occur, is a problem we do not deal with in this paper).

To begin, in all Figs. 4.1—4.4:

- In all subfigures, the horizontal axis depicts the progression of the coalition size.
- *Capacity subfig.* On the first graph of each figure, the capacity of the coalition is displayed. We can see how it is increased by selecting the appropriate agents until the goal (horizontal line) is reached.
- *Reliability subfig.* The second graph displays the mean reliability of our coalition.
- *Discharge subfig.* The third and last graph displays the discharge rate of the coalition. The goal of $1,000$ kW is shown as a horizontal line.

Heuristic Algorithm As explained in Section 3.1.5, this algorithm attempts (in a rather "greedy" manner) to identify the best EVs from the hypergraph. As we can observe in Fig. 4.1, it takes on average only 58.5 vehicles to reach the goal requirements, which is the most efficient use of resources observed across all our methods. The reliability achieved is also high, reaching a value of more than 1.5. We remind the reader that the mean reliability of

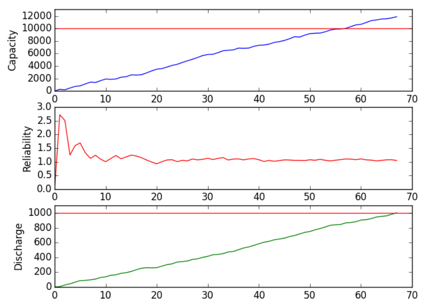

FIGURE 4.3: Coalition formation with the Minimal Transversal Algorithm

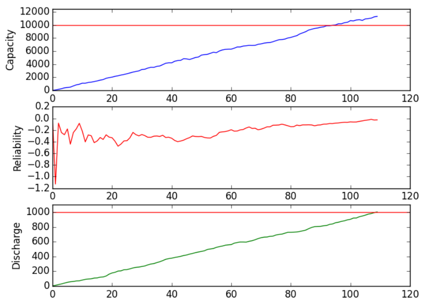

FIGURE 4.4: Coalition creation with the Simple Sampling Algorithm

Algorithm	Heuristic	Clustering	Transversal	Simple Sampling
Mean Coalition Size (# EVs)	58.5	98	64	109.3
Mean Running Time (ms)	25	709	120	24
Mean Generat.+Run. Time (ms)	1041	1725	1136	1040

TABLE 4.2: Summarizing the performance results

our pool of EVs is 0. This approach is also the most time and memory efficient of all. Specifically the algorithms average completion time is only $25ms$ for these experiments, and it also scales linearly into the millions as seen in Fig. 4.7 below.

Clustering Algorithm This method performs clustering, as explained in Section 3.1.4, and then takes random samples from the best cluster. Fig. 4.2 depicts its performance when using $k = 3$ clusters. Unfortunately, we cannot control how exactly the clusters are formed, so we do not have a guarantee that high quality vehicles will be clustered together. This leads to a mediocre result with an increased average coalition size, and a slightly-over-the-average reliability. The average size of coalitions meeting both requirements is 98. The average time required for the method's completion is 709ms. In Section 4.1.4, we showed how different k values affected our results.

Transversal Using the transversal algorithm and taking nodes from a list of minimal hitting set. Fig. 4.3 shows its performance. The transversal algorithm appears to work quite well since the average coalition size is only 64, slightly higher than that achieved by the heuristic approach. The reliability of the coalition is high, reaching values over 1.1. It can also scale quite well, reaching thousands of vehicles (cf. Fig. 4.5), but not as well as the heuristic approach. The average time to completion was $120ms$.

Simple Sampling Fig. 4.4 depicts our results for the Simple Sampling method. The average coalition size achieved with this algorithm is 109.3. The average completion time was 24ms. As expected, this algorithm achieves the weakest results among all our algorithms. The algorithms' performance is summarized in Table 4.2 for convenience.

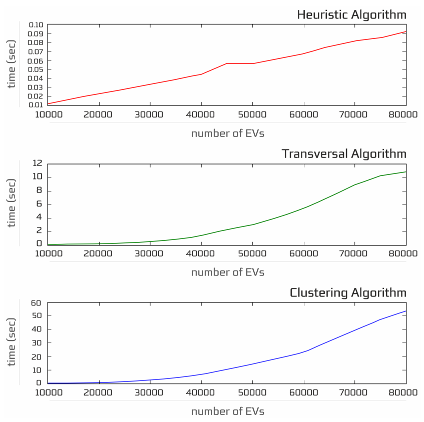

FIGURE 4.5: Scaling against an increasing EV population

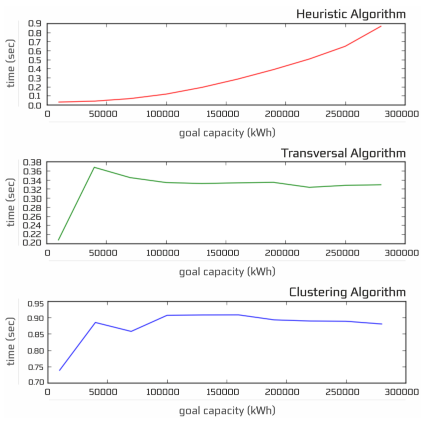

FIGURE 4.6: Scaling against an increasing "capacity" goal

4.1.3 Scaling Behaviour

We now test the scaling behaviour of our algorithms. First, we show how our algorithms scale with time when the *capacity* goal is increased. Then, we show how they scale as the number of EVs under consideration increases.

In Fig. 4.6 we can see how the transversal, heuristic and clustering algorithm scale against an increasing capacity goal (assuming any other goal remains fixed). The starting size of the available agents was kept constant at $20,000$ EVs for this experiment. We observe that the scaling behaviour of the heuristic algorithm against an increasing capacity goal is exponential. Nevertheless, its total required execution time is low, since it takes the algorithm 0.9 seconds to reach the goal capacity of $300,000$ kWh. The transversal algorithm scales with steps. The main reason for this is that the minimal transversal sets are generated before we select the agents of a coalition. If a minimal transversal set does not achieve the goal capacity, we generate a new one with more agents, till we reach the set capacity goal. This generates a step pattern, the first stages of which are shown in Fig. 4.6. In Fig. 4.6 we actually manage to see only one step because generating the minimal transversals with 3 EVs is enough to find good coalitions for all goals from $40,000$ kWh onwards (while it was enough to generate the minimal transversals with 2 EVs to cover the $10,000$ kWh capacity goal).

Now, the running time of the hypegraph clustering algorithm is largely independent of the size of the stated capacity goal. This is because the clustering itself, which is the part of the algorithm that requires the most processing power, takes place regardless of the final coalition requirements. Indeed, we observe in Fig. 4.6 that after an initial jump due to increased sampling requirements (cf. lines 6—8, Alg. 3) when moving from a goal of $10,000$ to $40,000$ kWh, the algorithm's running time remains largely unaltered.

Fig. 4.5 displays scaling against the initial EV population. The coalition goals were kept constant, and the same for all algorithms. The heuristic algorithm shows a linear scaling in time as the agent size grows. Specifically, the heuristic algorithm can scale *up to a million agents* within an acceptable time.

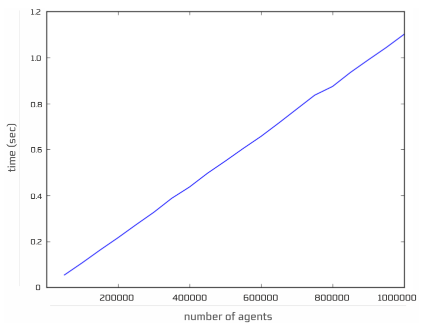

FIGURE 4.7: Scaling of the Heuristic Algorithm

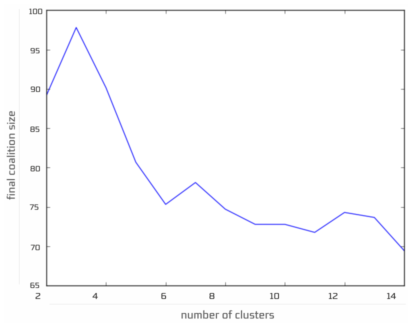

FIGURE 4.8: Evolution of the average size of coalitions produced with the Hypergraph Clustering method, when varying the number of clusters

Fig. 4.7 demonstrates this behaviour, starting from $50,000$ EVs. Of course, one expects that when the population reaches several millions, the complexity of the sorting algorithm will kick in, creating bottlenecks. Regardless, the fact that linear scalability is maintained up to $1,000,000$ agents is reassuring. By contrast, looking at Fig. 4.5, we observe that the transversal and clustering algorithms scale exponentially.

4.1.4 Varying the number of hypergraph clusters

We test our clustering algorithm further by modifying the number of clusters, k, since this is a parameter that can be optimized empirically, as explained in Section 3.1.4.

Fig. 4.8 displays the relation between k and the average coalition size that results from the clustering method (and which achieves the set goals). Creating a larger number of clusters results in smaller, and thus better, coalitions. Regardless, even when $k = 15$, the clustering algorithm still produces coalitions with more EVs than the heuristic one.

Appendix A

Appendix Title Here

Write your Appendix content here.

Bibliography

[1] Filippo Bistaffa et al. "Anytime coalition structure generation on synergy graphs". In: *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems. 2014, pp. 13–20.

[2] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. "Computational aspects of cooperative game theory". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 5.6 (2011), pp. 1–168.

[3] Georgios Chalkiadakis, Gianluigi Greco, and Evangelos Markakis. "Characteristic function games with restricted agent interactions: Core-stability and coalition structures". In: *Artificial Intelligence* 232 (2016), pp. 76–113.

[4] Georgios Chalkiadakis, Evangelos Markakis, and Nicholas R Jennings. "Coalitional stability in structured environments". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems. 2012, pp. 779–786.

[5] Filippos Christianos and Georgios Chalkiadakis. "Employing Hypergraphs for Efficient Coalition Formation with Application to the V2G Problem". In: *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands*. 2016, pp. 1604–1605.

[6] Thomas Eiter and Georg Gottlob. "Identifying the minimal transversals of a hypergraph and related problems". In: *SIAM Journal on Computing* 24.6 (1995), pp. 1278–1304.

[7] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28.1 (1979), pp. 100–108.

[8] Tackseung Jun and Jeong-Yoo Kim. "Hypergraph formation game". In: *Hitotsubashi Journal of Economics* (2009), pp. 107–122.

[9] Sachin Kamboj, Willett Kempton, and Keith S Decker. "Deploying power grid-integrated electric vehicles as a multi-agent system". In: *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2011, pp. 13–20.

[10] Sachin Kamboj et al. "Exploring the formation of electric vehicle coalitions for vehicle-to-grid power regulation". In: *AAMAS workshop on agent technologies for energy systems (ATES 2010)*. 2010.

[11] Gabriel de Oliveira Ramos, Juan C. Burguillo, and Ana LC Bazzan. "Dynamic constrained coalition formation among electric vehicles". In: *Journal of the Brazilian Computer Society* 20.1 (2014), pp. 1–15. ISSN: 1678-4804. DOI: 10.1186/1678-4804-20-8. URL: http://dx.doi.org/10.1186/1678-4804-20-8.

[12] Talal Rahwan et al. "An anytime algorithm for optimal coalition structure generation". In: *Journal of Artificial Intelligence Research (JAIR)* 34.1 (2009), pp. 521–567.

[13] Sarvapali D Ramchurn et al. "Putting the'smarts' into the smart grid: a grand challenge for artificial intelligence". In: *Communications of the ACM* 55.4 (2012), pp. 86–97.

[14] Tuomas Sandholm et al. "Coalition structure generation with worst case guarantees". In: *Artificial Intelligence* 111.1 (1999), pp. 209–238.

[15] Onn Shehory and Sarit Kraus. "Methods for task allocation via agent coalition formation". In: *Artificial Intelligence* 101.1 (1998), pp. 165–200.

[16] Konstantina Valogianni et al. "Effective management of electric vehicle storage using smart charging". In: *Proceedings of 28th AAAI Conference on Artificial Intelligence*. 2014, pp. 472–478.

[17] Meritxell Vinyals et al. "Stable Coalition Formation Among Energy Consumers in the Smart Grid". In: *Proceedings of the 2012 International Workshop on Agent Technologies for Energy Systems*. International Foundation for Autonomous Agents and Multi-Agent Systems. 2012.

[18] Thomas Voice, Sarvapali D Ramchurn, and Nicholas R Jennings. "On coalition formation with sparse synergies". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2012, pp. 223–230.

[19] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. "Learning with hypergraphs: Clustering, classification, and embedding". In: *Advances in Neural Information Processing Systems*. 2006, pp. 1601–1608.

[20] Yair Zick, Georgios Chalkiadakis, and Edith Elkind. "Overlapping coalition formation games: Charting the tractability frontier". In: *Proceedings of the 11th International Conference on AAMAS-2012-Volume 2*. 2012, pp. 787–794.